WHAT IS CLAIMED IS:

1. A data processing system comprising:

a first data bus for transmitting data requests at a first speed;

a second data bus for transmitting I/O data at a second speed; and

a non-blocking load buffer connected to said first and second data buses for holding said data requests and said I/O data so that a plurality of memory or I/O transactions may be performed simultaneously.

2. A data processing system according to Claim 1, wherein said plurality of memory or I/O transactions comprise loads.

3. A data processing system according to Claim 1, wherein said plurality of memory or I/O transactions comprise stores.

4. A data processing system according to Claim 1, wherein said second speed of said second data bus is slower than said first speed of said first data bus.

5. A data processing system according to Claim 1, wherein said first data bus is connected to a caching unit and a plurality of processors.

6. A data processing system according to Claim 5, wherein said second data bus is connected to a plurality of peripheral devices and/or memories.

7. A data processing system according to Claim 1, wherein said non-blocking load buffer comprises a control block and a memory array.

8. A data processing system according to Claim 1, wherein said non-blocking load buffer comprises a control block and a plurality of memory arrays.

9. A data processing system according to Claim 7, wherein said control block comprises a plurality of queues and data pools.

10. A data processing system according to Claim 9, wherein said plurality of queues comprise a pending queue corresponding to each of a plurality of peripheral devices and/or memories connected to said second data bus.

11. A data processing system according to Claim 7, wherein said non-blocking load buffer holds entries of said data requests and said I/O data in said queues and data pools, which include pending queues, and a return queue, an outstanding pool and a free pool.

12. A data processing system according to Claim 6, wherein load addresses are stored in a memory array of said non-blocking load buffer and sent to one of said plurality of peripheral devices and/or memories at said second speed, and then said load addresses returned from the one of said peripheral devices and/or memories are stored in said memory array until one of said processors is ready to receive the data.

13. A data processing system according to Claim 6, wherein said non-blocking load buffer comprises a memory array for storing both store addresses and data before being sent to one of said plurality of peripheral device and/or memories.

14. A data processing system according to Claim 5, wherein said caching unit comprises one cache.

15. A data processing system according to Claim 5, wherein said caching unit comprises a plurality of caches.

16. A data processing system according to Claim 15, wherein each of said caches corresponds to each of said processors.

17. A data processing system according to Claim 1, wherein said first data bus is connected to a caching unit and a processor.

18. A data processing system according to Claim 17, wherein said caching unit comprises one cache.

19. A data processing system according to Claim 17, wherein said caching unit comprises a plurality of caches.

20. A data processing system comprising:

a caching unit for storing data;

a plurality of processors for processing data;

a non-blocking load buffer for holding data requests and I/O data;

a high speed processor/cache bus for connecting said cache, said processors and said non-blocking load buffer and transmitting data there between at a first high speed;

a plurality of peripheral devices and/or memories for receiving and processing the I/O data; and

a peripheral bus for connecting said non-blocking load buffer and said plurality of peripheral devices and/or memories and transmitting data therebetween at a second speed slower than said first high speed, said non-blocking load buffer holding data requests and said I/O data while simultaneously performing a plurality of loads so that a continuous maximum throughput from said plurality of peripheral devices and/or memories is achieved.

21. A data processing system according to Claim 20, wherein said peripheral bus has a bandwidth at least equal to a peak of an input and an output bandwidth for said plurality of peripheral devices and/or memories and said non-blocking load buffer has a bandwidth at least equal to the sum of the bandwidth of said peripheral bus and bandwidth of said high speed processor/cache bus.

22. A data processing system according to Claim 20, wherein said data requests comprise an address, control information and data for each of a plurality of entries held in said non-blocking load buffer.

23. A data processing system according to Claim 22, wherein said control information includes request type, priority, data size, coherency, and load or store information for each entry.

24. A data processing system according to Claim 20, wherein said caching unit comprises one cache.

25. A data processing system according to Claim 20, wherein said caching unit comprises a plurality of caches.

26. A data processing system according to Claim 25, wherein each of said caches corresponds to each of said processors.

27. A non-blocking load buffer for a multi-processor system running real-time processes comprising:

a memory array for storing data; and

a control block for simultaneously performing a plurality of memory or I/O transactions.

28. A non-blocking load buffer according to Claim 27, wherein said plurality of memory or I/O transactions comprise loads.

29. A non-blocking load buffer according to Claim 27, wherein said plurality of memory or I/O transactions comprise stores.

30. A non-blocking load buffer according to Claim 27, wherein control block comprises a plurality of queues for temporarily storing data.

31. A non-blocking load buffer according to Claim 30, wherein said control block comprises pointers to entries in said memory array are queued.

32. A non-blocking load buffer according to Claim 30, wherein a plurality of independent pending queues correspond to each of a plurality of peripherals connected to the non-blocking load buffer.

33. A non-blocking load buffer according to Claim 27, wherein the data stored in said memory array comprises a plurality of entries each including address, data and control information.

34. A non-blocking load buffer according to Claim 27, wherein said memory array is divided into a plurality of portions including address/control memory array portions and data memory array portions.

35. A method for processing data comprising the steps of:

transmitting data requests at a first speed over a first data bus;

transmitting I/O data at a second speed slower than said first speed over a second data bus; and

holding said data requests and said I/O data in a non-blocking load buffer connected to said first and second data buses so that a plurality of memory or I/O transactions may be performed simultaneously.

36. A method for processing data according to Claim 35, wherein said plurality of memory or I/O transactions comprise loads.

37. A method for processing data according to Claim 35, wherein said plurality of memory or I/O transactions comprise stores.

38. A method for processing data according to Claim 36, further comprising the steps of:

initializing each of a plurality of entries in said non-blocking load buffer to be free;

issuing a non-blocking load by one of a plurality of processors to said first data bus;

selecting one of said plurality of entries in said non-blocking load buffer as a pending entry, writing an address of said non-blocking load therein, and allocating space in said entries for returned data;

writing said pending entry to one of a plurality of peripheral devices and/or memories connected to said second data bus as an outstanding entry when the one peripheral is ready to accept data;

writing data returned from the one peripheral to the allocated space; and

reading data from the allocated space, returning the read data to a processor connected to said first data bus and freeing the entry in said non-blocking load buffer.

add A1